



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/777,954	02/07/2001	Hideaki Minamide	401073	5794

23548 7590 03/28/2006

LEYDIG VOIT & MAYER, LTD  
700 THIRTEENTH ST. NW  
SUITE 300  
WASHINGTON, DC 20005-3960

EXAMINER

KISS, ERIC B

ART UNIT	PAPER NUMBER
----------	--------------

2192

DATE MAILED: 03/28/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

**Office Action Summary**

Application No.

09/777,954

Applicant(s)

MINAMIDE ET AL.

Examiner

Eric B. Kiss

Art Unit

2192

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1) ☒ Responsive to communication(s) filed on 23 December 2005.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

- 4) ☒ Claim(s) 2, 4, 5, 7-12, 14-19 and 26-28 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 2, 4, 5, 7-12, 14-19 and 26-28 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

**Application Papers**

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
  2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)  
Paper No(s)/Mail Date \_\_\_\_\_
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date. \_\_\_\_\_
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: \_\_\_\_\_

### **DETAILED ACTION**

1. The reply filed 23 December 2005 has been received and entered. Claims 2, 4, 5, 7-12, 14-19, and 26-28 are pending.

#### ***Information Disclosure Statement***

2. As noted previously (Non-final Rejection mailed 07/26/2005 at 2-3,) the information disclosure statement filed 7 February 2001 fails to comply with 37 CFR 1.98, and accordingly, the cited non-patent document has not been considered.

#### ***Response to Amendment***

3. The rejection under 35 U.S.C. § 112, second paragraph, is withdrawn in view of Applicant's amendments to the claims.

#### ***Response to Arguments***

4. Applicant's arguments filed 23 December 2005 have been fully considered but they are not persuasive.

In response to Applicant's arguments on p. 9, last paragraph, continuing through the middle of p. 10, it is noted that amended claim 2 does not require optimization before the control program is converted to a programming language.

In response to Applicant's arguments on p. 10, in paragraphs 4-5, the Examiner maintains that Kim discloses an estimation of execution time. This is apparent by the use of mean execution times of seven classes of LD mnemonics in the execution time equation. See Equation 2 and Figure 7. Chester is not relied upon for monthly payments and loan balances as Applicant suggests, but rather Chester is relied upon as teaching the known use of indexed data structures (i.e., tables) to evaluate a formula, such as Equation 2 of Kim, involving indexed data. Kim

Art Unit: 2192

further discloses determining and selecting the sample program most similar to the control program to estimate processing time (for example, the time calculation depends on the parameter P\_basic, which represents the percentage of basic mnemonics, and is changed according to the application program. This parameter, as reflected in Equation 2, adjusts the specific weighting of known mnemonic execution times from Figure 7, thus adjusting the execution time calculation to better reflect actual execution time).

In response to Applicant's arguments on p. 11, in paragraphs 1-2, Applicant's argument that the LD block of Kim is not a workable part of a ladder program is unpersuasive. Fig. 4(a) of Kim clearly illustrates such a ladder program part.

In response to Applicant's arguments on p. 11, in paragraph 4, as disclosed, for example, in the second column of p. 3, the PLC has a number of branches because it decides, by logic operation, whether it executes a rung or not. Thus, these rungs serve as jump destinations dependent on the result of the decision to execute the rungs.

In response to Applicant's arguments on p. 11, in paragraph 5, the introduction to section 2 of Kim describes clearly how PLCs interface with I/O devices through their programming. Further, as discussed above, Fig 4 demonstrates the relationship between ladder logic rungs and controllable blocks.

In response to Applicant's arguments on p. 13, in paragraphs 3-6, claim 19 recites instrumenting source code for debugging and relating the "execution part" of the control program to the source code through displayed line numbers. Rosenberg teaches such a mapping of source code to executable code through displayable line numbers. See "Context Is the Torch in a Dark Cave" on pp. 9-11 and "Source-level (Symbolic) versus Machine-level" on p 12.

Rosenberg further describes step execution. See, for example, the second paragraph of “Current State-of-the-Art” on p. 5.

***Claim Rejections - 35 USC § 102***

5. The text of those sections of Title 35, U.S. Code not included in this action can be found in a prior Office action.

6. Claims 5, 7-9, and 26 are rejected under 35 U.S.C. 102(b) as being anticipated by Hyung Seok Kim, et al., “A Translation Method of Ladder Diagram on PLC with Application to a Manufacturing Process,” 1999 (hereinafter [Kim99]).

As per claim 5, [Kim99] discloses a control-program-development supporting apparatus that develops a control program described with a sequential-control language, said control-program-development supporting apparatus comprising: a control-program dividing unit which divides the control program into a plurality of controllable blocks (see, for example, section 3 on pp. 2-4); and a compiler which compiles at least some of the controllable blocks into execution codes directly executable by a programmable controller (see, for example, section 3 on pp. 2-4). [Kim99] further discloses the programmable controller including a microprocessor having at least one cache (see, for example, section 3.2 on p. 4).

As per claim 7, [Kim99] further discloses the control program being a ladder logic diagram or an instruction list generated from the ladder diagram, and the control-program dividing unit dividing the control program into a plurality of controllable blocks at a

Art Unit: 2192

predetermined rung in the ladder diagram to generate a program file for every controllable block concerned (see, for example, section 3 on pp. 2-4).

As per claim 8, [Kim99] further discloses the control program being a ladder diagram or an instruction list generated from the ladder diagram, and the control-program dividing unit dividing the control program into a plurality of controllable blocks at a predetermined rung serving as a jump destination for a jump instruction in the ladder diagram to generate a program file for every controllable block (see, for example, section 3 on pp. 2-4).

As per claim 9, [Kim99] further discloses the control-program being a ladder diagram or an instruction list generated from the ladder diagram, and the control-program dividing unit extracting at least some rungs including instruction to a common input or output device from the ladder diagram, at least some of the rungs extracted constituting one controllable block, and generating a program file for every controllable block (see, for example, section 3 on pp. 2-4).

As per claim 26, [Kim99] discloses a storing unit which stores the execution codes; a microprocessor which includes a cache (an inherent feature of the TMS320C40 digital signal processor) and directly executing the execution codes (see, for example, section 3 on pp. 2-4); and a control-program-development supporting apparatus that develops a control program described with a sequential-control language, the control-program-development supporting apparatus having, a control-program dividing unit which divides the control program into a plurality of controllable blocks (see, for example, section 3 on pp. 2-4); and a compiler which compiles at least some of the controllable blocks into execution codes directly executable by a programmable controller (see, for example, section 3 on pp. 2-4).

***Claim Rejections - 35 USC § 103***

7. The text of those sections of Title 35, U.S. Code not included in this action can be found in a prior Office action.

8. Claims 4 and 11 are rejected under 35 U.S.C. 103(a) as being unpatentable over [Kim99] in view of *Mastering Excel 97*, 4<sup>th</sup> ed., 1997, by Thomas Chester and Richard H. Alden (hereinafter [ChA97]).

As per claim 4, [Kim99] discloses a control-program-development supporting apparatus that develops a control program described with a sequential-control language, said control-program-development supporting apparatus comprising a compiler which compiles the control program into codes directly executable by a microprocessor that includes an acceleration unit (see, for example, section 3 on pp. 2-4). [Kim99] further discloses a processing-time rough-estimating unit which relates a sample program having a known processing time with the control program corresponding to the execution codes to estimate sequential-processing execution time of a programmable controller (see, for example, section 4 on pp. 4-5; the use of underlying apparatus components is inherent in determining the execution times of the modified programs as disclosed; [Kim99] further appears to disclose the use of the TMS320C40 digital signal processor for such a purpose, as disclosed, for example, in section 3.2 on p. 4). [Kim99] further discloses determining and selecting the sample program most similar to the control program to estimate processing time (for example, the time calculation depends on the parameter P<sub>basic</sub>, which represents the percentage of basic mnemonics, and is changed according to the application program. This parameter, as reflected in Equation 2, adjusts the specific weighting of known

mnemonic execution times from Figure 7, thus adjusting the execution time calculation to better reflect actual execution time). [Kim99] fails to expressly disclose the use of a relating table in implementing such relating. However, the calculated execution time equation (equation (2) on p. 5) contains indexed values (*e.g.*,  $T_i$ , where  $i$  is an index from 1 to 6) related to the execution times of various ladder diagram mnemonics (see Figure 7). As is well known in the computer programming art, a table is an indexed data structure, allowing data to be retrieved and processed based on a unique index. An example of the use of a table (for example, a spreadsheet) in performing a calculation can be found in [ChA97] (see, for example, the loan calculator described on pp. 114-117). It would have been obvious to one of ordinary skill in the computer art at the time the invention was made to include the use of such a table in the execution time calculation of [Kim99] as tables are well suited to such tasks involving calculations performed on indexed data.

As per claim 11, in addition to the disclosure applied to claim 5, [Kim99] further discloses a processing-time rough-estimating unit which relates a sample program having a known processing time with the control program corresponding to the execution codes to estimate sequential-processing execution time of a programmable controller (see, for example, section 4 on pp. 4-5; the use of underlying apparatus components is inherent in determining the execution times of the modified programs as disclosed; [Kim99] further appears to disclose the use of the TMS320C40 digital signal processor for such a purpose, as disclosed, for example, in section 3.2 on p. 4). [Kim99] further discloses determining and selecting the sample program most similar to the control program to estimate processing time (for example, the time calculation depends on the parameter  $P\_basic$ , which represents the percentage of basic

Art Unit: 2192

mnemonics, and is changed according to the application program. This parameter, as reflected in Equation 2, adjusts the specific weighting of known mnemonic execution times from Figure 7, thus adjusting the execution time calculation to better reflect actual execution time). [Kim99] fails to expressly disclose the use of a relating table in implementing such relating. However, the calculated execution time equation (equation (2) on p. 5) contains indexed values (*e.g.*,  $T_i$ , where  $i$  is an index from 1 to 6) related to the execution times of various ladder diagram mnemonics (see Figure 7). As is well known in the computer programming art, a table is an indexed data structure, allowing data to be retrieved and processed based on a unique index. An example of the use of a table (for example, a spreadsheet) in performing a calculation can be found in [ChA97] (see, for example, the loan calculator described on pp. 114-117). It would have been obvious to one of ordinary skill in the computer art at the time the invention was made to include the use of such a table in the execution time calculation of [Kim99] as tables are well suited to such tasks involving calculations performed on indexed data.

9. Claims 12, 14-16, and 27 are rejected under 35 U.S.C. 103(a) as being unpatentable over [Kim99] in view of U.S. Patent No. 5,504,902 to McGrath et al.

As per claim 12, [Kim99] discloses a control-program-development supporting apparatus that develops a control program described with a sequential-control language, said control-program-program-development supporting apparatus comprising: a control-program dividing unit which divides the control program into a plurality of controllable blocks (see, for example, section 3 on pp. 2-4); a control-program converting unit which converts at least some of the controllable blocks into advanced-language control programs described with a computer-

Art Unit: 2192

readable advanced language for every controllable block (see, for example, section 3 on pp. 2-4); and a compiler which compiles at least some of the computer-readable advanced programming languages corresponding to every controllable block into codes directly executable by a programmable controller (see, for example, section 3 on pp. 2-4). [Kim99] fails to expressly disclose the “advanced language” being a high-level language. However, McGrath et al. teaches such a conversion from a ladder-based language to a high-level language (see, for example, col. 4, line 59, through col. 5, line 17). Therefore, it would have been obvious to one of ordinary skill in the computer art at the time the invention was made to modify the conversion of [Kim99] to including such a high-level language. One would be motivated to do so to provide an additional means to edit and debug a control program.

As per claims 14-16, see the disclosure applied above to claims 5 and 7-9. For reasons stated above, such claims also would have been obvious.

As per claim 27, [Kim99] discloses a storing unit which stores the execution codes (see, for example, section 3 on pp. 2-4); a microprocessor which includes an acceleration unit and directly executing the execution codes (see, for example, section 3 on pp. 2-4); and a control-program-development supporting apparatus that develops a control program, described with a sequential-control language, the control-program-development supporting apparatus having, a control-program dividing unit which divides the control-program into a plurality of controllable blocks (see, for example, section 3 on pp. 2-4); a control-program converting unit which converts at least some of the controllable blocks into advanced-language control programs described with a universal-computer-readable advanced language for every controllable block (see, for example, section 3 on pp. 2-4); and a compiler which compiles at least some of universal-computer-

Art Unit: 2192

readable advanced programming languages corresponding to every controllable block into codes directly executable by a programmable controller (see, for example, section 3 on pp. 2-4).

[Kim99] fails to expressly disclose the “advanced language” being a high-level language.

However, McGrath et al. teaches such a conversion from a ladder-based language to a high-level language (see, for example, col. 4, line 59, through col. 5, line 17). Therefore, it would have been obvious to one of ordinary skill in the computer art at the time the invention was made to modify the conversion of [Kim99] to including such a high-level language. One would be motivated to do so to provide an additional means to edit and debug a control program.

10. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over [Kim99] in view of *McGrath et al.*, as applied to claim 12 above, and further in view of [ChA97].

As per claim 18, in addition to the disclosure applied to claim 12, [Kim99] further discloses a processing-time rough-estimating unit which relates a sample program having a known processing time with the control program corresponding to the execution codes to estimate sequential-processing execution time of a programmable controller (see, for example, section 4 on pp. 4-5; the use of underlying apparatus components is inherent in determining the execution times of the modified programs as disclosed; [Kim99] further appears to disclose the use of the TMS320C40 digital signal processor for such a purpose, as disclosed, for example, in section 3.2 on p. 4). [Kim99] further discloses determining and selecting the sample program most similar to the control program to estimate processing time (for example, the time calculation depends on the parameter P<sub>basic</sub>, which represents the percentage of basic

mnemonics, and is changed according to the application program. This parameter, as reflected in Equation 2, adjusts the specific weighting of known mnemonic execution times from Figure 7, thus adjusting the execution time calculation to better reflect actual execution time). [Kim99] fails to expressly disclose the use of a relating table in implementing such relating. However, the calculated execution time equation (equation (2) on p. 5) contains indexed values (*e.g.*,  $T_i$ , where  $i$  is an index from 1 to 6) related to the execution times of various ladder diagram mnemonics (see Figure 7). As is well known in the computer programming art, a table is an indexed data structure, allowing data to be retrieved and processed based on a unique index. An example of the use of a table (for example, a spreadsheet) in performing a calculation can be found in [ChA97] (see, for example, the loan calculator described on pp. 114-117). It would have been obvious to one of ordinary skill in the computer art at the time the invention was made to include the use of such a table in the execution time calculation of [Kim99] as tables are well suited to such tasks involving calculations performed on indexed data.

11. Claims 2 and 10 are rejected under 35 U.S.C. 103(a) as being unpatentable over [Kim99] in view of Alfred V. Aho, et al., "Compilers: Principles, Techniques, and Tools," 1988 (hereinafter [Aho88]).

As per claim 2, [Kim99] discloses a control-program-development supporting apparatus that develops a control program described with a sequential-control language, said control-program-development supporting apparatus comprising a compiler which compiles the control program into codes directly executable by a microprocessor that includes an acceleration unit

Art Unit: 2192

(see, for example, section 3 on pp. 2-4). [Kim99] further discloses an optimization filtering unit which reconstructs the control program into an optimum code system by rearranging codes for locally arranging instructions for a common input or output device, wherein a control program optimized by said optimization filtering unit is newly used as the control program (see, for example, section 3 on pp. 2-4). [Kim99] fails to expressly disclose excluding not-cited variables and redundant codes. However, [Aho88] teaches such well-known compiler optimization techniques as dead/redundant/unreachable code elimination (see, for example, section 9.9 on pp. 554-557; and section 10.2 on pp. 592-598). Therefore, it would have been obvious to one of ordinary skill in the computer art at the time the invention was made to modify the optimizing compiler framework disclosed by [Kim99] to include such known optimization techniques as taught by [Aho88]. One would be motivated to do so to improve the quality of resulting generated code.

As per claim 10, in addition to the disclosure applied above to claim 5, [Kim99] discloses a control-program-development supporting apparatus that develops a control program described with a sequential-control language, said control-program-development supporting apparatus comprising a compiler which compiles the control program into codes directly executable by a microprocessor that includes an acceleration unit (see, for example, section 3 on pp. 2-4). [Kim99] further discloses an optimization filtering unit which reconstructs the control program into an optimum code system by rearranging codes for locally arranging instructions for a common input or output device, wherein a control program optimized by said optimization filtering unit is newly used as the control program (see, for example, section 3 on pp. 2-4). [Kim99] fails to expressly disclose excluding not-cited variables and redundant codes. However,

Art Unit: 2192

[Aho88] teaches such well-known compiler optimization techniques as dead/redundant/unreachable code elimination (see, for example, section 9.9 on pp. 554-557; and section 10.2 on pp. 592-598). Therefore, it would have been obvious to one of ordinary skill in the computer art at the time the invention was made to modify the optimizing compiler framework disclosed by [Kim99] to include such known optimization techniques as taught by [Aho88]. One would be motivated to do so to improve the quality of resulting generated code.

12. Claim 17 is rejected under 35 U.S.C. 103(a) as being unpatentable over [Kim99] and *McGrath et al.*, as applied to claim 12 above, in view of [Aho88].

As per claim 17, in addition to the disclosure and teachings applied above to claim 12, [Kim99] discloses an optimization filtering unit which reconstructs the control program into an optimum code system by recombining logical operations and rearranging codes for locally arranging instructions for a common input or output device, wherein a control program optimized by said optimization filtering unit is newly used as the control program (see, for example, section 3 on pp. 2-4). [Kim99] fails to expressly disclose excluding not-cited variables and redundant codes. However, [Aho88] teaches such well-known compiler optimization techniques as dead/redundant/unreachable code elimination (see, for example, section 9.9 on pp. 554-557; and section 10.2 on pp. 592-598). Therefore, it would have been obvious to one of ordinary skill in the computer art at the time the invention was made to modify the optimizing compiler framework disclosed by [Kim99] to include such known optimization techniques as taught by [Aho88]. One would be motivated to do so to improve the quality of resulting generated code.

13. Claims 19 and 28 are rejected under 35 U.S.C. 103(a) as being unpatentable over [Kim99] in view of *McGrath et al.* and Jonathan B. Rosenberg, "How Debuggers Work: Algorithms, Data Structures, and Architecture," 1996 (hereinafter [Ros96]).

As per claims 19 and 28, [Kim99] discloses a storing unit which stores execution codes (see, for example, section 3 on pp. 2-4); a microprocessor including a cache (see, for example, section 3.2 on p. 4; a cache is an inherent feature of the TMS320C40 digital signal processor) and directly executing the execution codes (see, for example, section 3 on pp. 2-4); and a control-program-development supporting apparatus that develops a control program described with a sequential-control language (see, for example, section 3 on pp. 2-4), having a control-program converting unit which converts a control program into an advanced-programming-language control program described with a computer-readable advanced programming language (see, for example, section 3 on pp. 2-4).

[Kim99] fails to expressly disclose the "advanced language" being a high-level language. However, *McGrath et al.* teaches such a conversion from a ladder-based language to a high-level language (see, for example, col. 4, line 59, through col. 5, line 17). Therefore, it would have been obvious to one of ordinary skill in the computer art at the time the invention was made to modify the conversion of [Kim99] to including such a high-level language. One would be motivated to do so to provide an additional means to edit and debug a control program.

[Kim99] fails to expressly disclose a debugging-code generating unit which generates a debugging control program by inserting a line number into a part corresponding to each line,

Art Unit: 2192

constituting the instruction list in source codes, constituting the advanced-programming-language control program; and a debugging executing unit which displays each line of the instruction list and the execution part of the advanced-programming-language control program by relating the former with the latter. However, [Ros96] teaches that debuggers are critical tools for software development (see, for example, p. 1, line 1). [Ros96] further teaches that showing source code line correspondence is part of the most important information the programmer needs during debugging, namely program context information (see, for example, "Context Is the Torch in a Dark Cave" on pp. 9-11) and that the developer of an application who is using a debugger has a lot to gain if the original source code is mapped directly to the application's machine code (see, for example, "Source-level (Symbolic) versus Machine-level" on p 12). Therefore, it would have been obvious to one of ordinary skill in the computer art at the time the invention was made to modify the system of [Kim99] to include such program code line number correspondence as per the teachings of [Ros96]. One would be motivated to do so to allow for efficient debugging of code under development.

### *Conclusion*

14. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

15. Any new ground(s) of rejection presented in this Office action were necessitated by Applicant's amendment. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

16. Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Eric B. Kiss whose telephone number is (571) 272-3699. The Examiner can normally be reached on Tue. - Fri., 7:00 am - 4:30 pm. The Examiner can also be reached on alternate Mondays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Tuan Dam, can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.


Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Art Unit: 2192

Any inquiry of a general nature should be directed to the TC 2100 Group receptionist:

571-272-2100.

EBK /EBK  
March 19, 2006



TUAN DAM  
SUPERVISORY PATENT EXAMINER